Restful Objects 1.0: Developers' Guide

Framework Development Manual

Version Version 0.1

Preface

Restful Objects is a sister project for the Naked Objects framework, providing a REST web service for your domain objects.

This developers' guide explains how to build *Restful Objects* from source, allowing you to contribute back and extend the range of capabilities. If you are simply interested in using *Restful Objects* as-is, please consult the user guide.

Restful Objects is hosted on SourceForge, and is licensed under Apache Software License v2. *Naked Objects* is also hosted on SourceForge, and is also licensed under Apache Software License v2.

Chapter 1

Introduction

This chapter introduces the organization of this developers' guide.

Restful Objects is one of a number of sister projects for Naked Objects. Each of these sister projects are organized along the same general lines: they have the same directory structure, the same coding conventions, a shared "corporate" Maven POM to define build artifacts, the same release process and so on.

The Star Objects project is an umbrella for all of these sister projects. As such it holds the corporate POM and a number of other shared artifacts, such as a site template so that the Maven sites for all sister projects have the same general look-n-feel. It also hosts a Maven snapshot repository and release repository.

In addition, the *Star Objects* also has a developers guide (available online here). This describes how to build any given sister project from source, how to be a contributor, and how (as a project admin) to release code artifacts to the repositories and how to deploy the site.

This developers guide therefore provides only a high level outline of the structure of the modules, and provides only summary steps for how to build and deploy the sister projects. Any variations from the standard procedures described in the *Star Objects* developers guide are also given.

Tihs guide also provides design/implementation notes, in Chapter 4, *Design Notes*. If you are thinking about or fixing a bug or contributing a new feature, you might find some starting points here (over and above reading the Javadocs, tests and code, of course).

Chapter 2

Modules

This chapter describes the modules that make up Restful Objects'.

The modules that make up *Restful Objects* follow the general conventions of sister projects, with a *main* module, a *support* module and a *testapp* module. You can read more about this in the *Star Objects* developer guide.

2.1. Directory Structure

The source code directory structure for *Restful Objects* is as follows:

```
trunk/
  main/               # main release for Restful, including Maven site
    applib/             # the Restful application library, for client-side code
    viewer/             # the Restful viewer implementation
    documentation/      # this documentation
  support/            # Support for application developers
    release/            # defines dependencies for projects using the 'viewer' submodule
    archetype/          # (placeholder)
  testapp/            # application for testing - not released
tags/
  main                # tags for trunk/main
  archetype           # tags for trunk/archetype
```

As is usual, to ensure that tags go into the correct location when releasing, the `mvn-release-plugin` plugin has been configured (using `<tagBase>`) to override its default location.

You can checkout the entire trunk using Subversion:

```
svn co https://restfulobjects.svn.sourceforge.net/svnroot/restfulobjects/trunk ~/
restfulobjects/trunk
```

2.2. Main Modules

As the above shows, there are two separate released artifacts:

The *main* (`org.starobjects.restful:main`) is a multimodule project that defines the main artifacts that implement *Restful Objects*. It contains:

- the *applib* (`org.starobjects.restful:applib`)

  The application library defines a set of interfaces representing the resources annotated using javax.ws.rs (JSR311) annotations. These in effect define the endpoints available for clients. The library also contains some helper utility classes.

  The applib uses `org.starobjects.restful:main` as its parent.

- the *viewer* (`org.starobjects.restful:viewer`)

  The viewer provides an implementation of the applib resources, exposing the domain objects as RESTful resources in XHTML 5. The resource include forms to allow testing from web browsers. To support browsers predating XHTML 5, Javascript is also served up (only Firefox 3.0 has been tested).

  The viewer also uses `org.starobjects.restful:main` as its parent.

- the *documentation* (`org.starobjects.restful:documentation`)

  The documentation submodule contains the user and developers' guides. It also uses `org.starobjects.restful:main` as its parent.

It uses the corporate POM (org.starobjects.star:corporate) as its parent.

2.3. Support Modules

The *support* release (`org.starobjects.restful:support`) is a multimodule project that provides artifacts to help application developers use *Restful Objects* in their own projects. It contains:

- the *release* module (`org.starobjects.restful:release`)

  This is a convenience module that can be used as a parent by projects using *Restful Objects*. Its primary purpose is to define a consistent set of versions in `<dependencyManagement>` tag.

  Note that this module does not inherit from the *support* POM, instead it inherits from the *Naked Objects Framework*'s equivalent `org.nakedobjects:release` module (thus defining a stack of dependencies).

- the *archetype* module

  The projects generated by the archetype [will] use the *release* module.

  > **Note**
  >
  > At the time of writing the archetype artifact is incomplete and is not included in the build.

Like *main*, the *support* module also uses the corporate POM (org.starobjects.star:corporate) as its parent.

2.4. TestApp Module

The testapp module is a test application for adhoc testing of FitNesse. It is not a released artifact.

Chapter 3

Building, Documenting and Deploying

This chapter outlines how to build, document and deploy Restful Objects.

The build, documentation and deployment process follows the general standard for sister projects, as documented in the *Star Objects* developers' guide. The sections in this chapter correspond to the parts one, two and three of *Star Objects* developers' guide.

3.1. Building from Source

There are no special steps required for building *Restful Objects* from source.

You can therefore just follow the processes described in *Star Objects* developers' guide:

- build the main:

```
$ cd ~/restfulobjects/trunk/main
$ mvn clean install
```

- build the support:

```
$ cd ~/restfulobjects/trunk/support
$ mvn clean install
```

3.2. Contributing Changes

There are no special considerations for contributing changes for *Restful Objects*. You can therefore just follow the processes described in *Star Objects* developers' guide.

3.3. Release Process

There are no special considerations for releasing/deploying for *Restful Objects*. You can therefore just follow the processes described in *Star Objects* developers' guide, to:

- for deployments, update `~/.m2/settings.xml`:

```
<servers>
  <server>
    <id>restfulobjects-site</id>
    <username>xxx</username>
    <password>xxx</password>
  </server>
</servers>
```

- make documentation changes to DocBook and to the site
- deploy the site locally

```
$ cd ~/restfulobjects/trunk/main
$ mvn site-deploy -D dist=local
```

  This will deploy to `/tmp/m2-sites/restfulobjects`.

- deploy a code snapshot

  First, deploy *main*:

```
$ cd ~/restfulobjects/trunk/main
$ mvn clean install deploy -D dist=remote
```

  Then, deploy *support*:

```
$ cd ~/restfulobjects/trunk/support
$ mvn clean install deploy -D dist=remote
```

- tag a release and then deploy a code release

TODO: details required here.

- deploy a site remotely

  then, deploy the site (you'll also need a sourceforge terminal session running; see *Star Objects* developers guide for details):

```
$ cd ~/restfulobjects/trunk/main
$ mvn site-deploy -D dist=remote
```

Chapter 4

Design Notes

This chapter will contain design notes on the implementation of Restful Objects.

TODO: describe the internal design/architecture here...